[Continued on next page]

(54) Title: CONTENT MANAGEMENT AND TRANSFORMATION SYSTEM FOR DIGITAL CONTENT
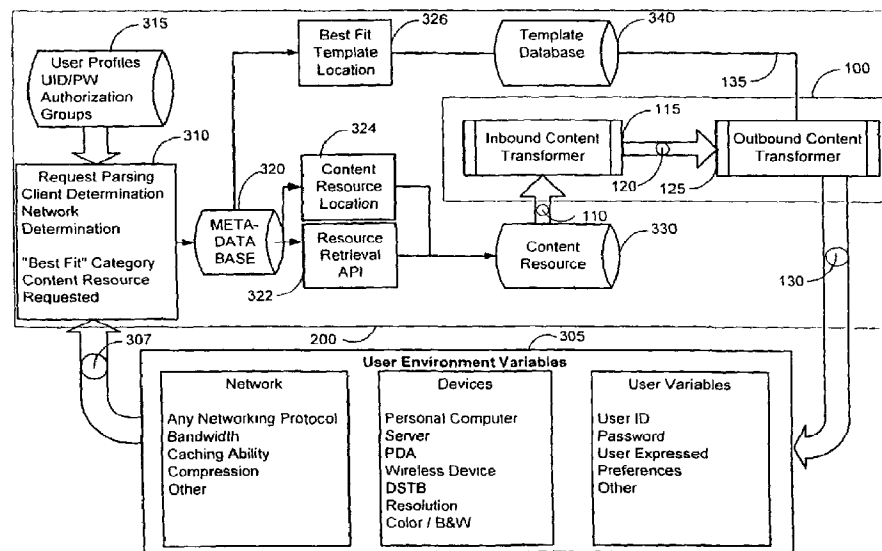
(57) Abstract: A transformation engine that enables content and information to be transformed from one format, a source format,
to a format that is compatible with the requesting device, a destination format. Advantageously, various device types can access and
share content via a network without concern as to the original format of the content. When a client device provides a request for
content, the transformation engine identifies delivery characteristics of the client device, and identifies a source for the requested
content. The transformation engine then transforms the source formatted content into a format identified through a best fit analysis
of the delivery characteristics of the device.

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii)) for all designations*

Published:

— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

CONTENT MANAGEMENT AND TRANSFORMATION
SYSTEM FOR DIGITAL CONTENT

5

TECHNICAL FIELD

10   The present invention relates to the transformation of information from one of a plurality of formats that may be utilized by an information source, to another of a plurality of formats that is required or desired by a destination for that information and, more particularly, the transformation of content from one markup language format to the same or another markup language format. In addition, the present

15   invention relates to use of this transformation in allowing a variety of platforms to access content and/or applications regardless of the source format of that content

BACKGROUND OF THE INVENTION

        One of the primary capabilities that has fueled the popularity of the Internet is

20   the ability to share information and technology. What other technology could a professor in a remote village in China use to share his ability to program or create content with a student in Midland, Texas? Through the technology of the Internet and its ability to allow downloading of applications, resources, and information using client-server principles, these types of transactions take place millions of times every

25   hour.

        Many times, during the spawning of a new technology, in an effort to rush the deployment and acceptance of the new technology, the technological community may adopt a less sophisticated or inferior embodiment of a new technology. Such compromises are usually driven by the consumer or the desire to have a product to

30   market before one's competitors. The adoption of inferior embodiments can result in

creating artificial limitations on the technology - limitations based on format and compatibility issues rather than technology issues. The rapid growth and deployment of the world wide web, coupled with the introduction of increasingly sophisticated client formats, languages, types of content and platforms, has given rise to such a

5   situation.

When the world is faced with an inferior embodiment of a new technology, over time, the problem usually is corrected. Two methods for correcting the problem include the deployment of a new or next generation technology, or the development of another technology to resolve or alleviate the problem. The present invention is

10   directed towards this second method, while at the same time, providing a revolutionary technology that will expand the usability of the Internet.

One of the primary roles of the Internet is as a source of content. Due to the ease of a device being interfaced to the Internet, a vast array of different devices are connected to and interact over the Internet. Each of these devices may be a source of

15   content to other devices through the Internet or, may attempt to access content from other devices through the Internet. Similar devices, or devices that store and utilize content in a similar format, can easily share content over the Internet. However, dissimilar devices, or devices that utilize different or even incompatible formats, cannot easily share content.

20   This information or content incompatibility problem is not unique to the Internet. This problem has also reared its head in other areas such as word processing, databases, or other applications that create and store files in a particular format. One technique that has been adopted is the use of conversion programs. A conversion program will read information in one format and convert it to a different format. For

25   instance, Microsoft Word includes an embedded conversion program that will allow a Word Perfect document to be converted to Microsoft Word format. This type of conversion program is not a viable solution for the sharing of content over the Internet. One reason is that there is a large number of formats, and even new formats being introduced, for content on the Internet. The conversion program would be very

30   complex, processor intensive, and large. In addition, when accessing content through

2

the Internet, it may be difficult or impossible to know in what format the content is stored. For small, portable, and wireless devices that access content over the Internet, these characteristics render this technique useless. Thus, there is a need in the art to allow devices that require content in a particular format, to access content through the

5   Internet, or any other local or global network, regardless of the format in which the content is stored.

One general format for content that is prolific on the Internet is markup languages. A markup language is a set of codes and/or characters that are interspersed with text and references to other content. The markup language directs the processing

10   of an application that displays the content. Many different markup languages have been developed and content can be found on the Internet in any one of the variety of markup languages. However, the basic function of a markup language is to instruct an application with regards to how to display or render a particular item. For instance, the following line in Rich Text Format, a markup language used by many personal

15   computer based applications,

{\b\f3\fs24\ul Melia Design Group - Creators of Nimbus\'99}

would result in causing a work processor to render the name "Melia Design Group - Creators of Nimbus™" as bolded (\b), 12 point (\fs24) courier font (\f3), and underlined with one line (\ul):

20   **Melia Design Group - Creators of Nimbus™**

The Standard Generalized Markup Language (SGML) is a descriptive markup language that basically provides a set of rules for creating other markup languages. Rather than instructing an application as to the particular aspects of a piece of text (i.e., bold, underlined, italicized) the SGML language describes a category for the text

25   (i.e., title, level 2 heading, bulleted entry, etc.). The exact particulars of the SGML format can be found in the standard ISO 8879:1986. Several markup languages have been created through SGML, the most popular of which is the Hyper Text Markup Language (HTML) which was mainly created for use on the world wide web. SGML has also been used to create markup languages for a multitude of applications

3

including word processors, parsers, and publishing, as well as other document formatting languages such as DHTML, XML, XHTML, GML and HDML.

The eXtensible Markup Language (XML) is an SGML markup language supported by the World Wide Web Consortium (W3C) and basically is a web-ready subset of SGML. As a subset of SGML, XML is more suitable for providing content to devices that are limited in memory or processing resources. Parsers and applications that utilize XML can be smaller and less processor intensive. As a subset of SGML, content that is stored in XML format is also valid SGML content. However, all SGML content is not necessarily valid XML content.

Thus, many markup languages have been developed (some of which are compatible with others, incompatible with others, limited in capabilities, etc.), and are in use within the Internet today. These markup languages allow devices and/or applications that support the markup language, to render content onto the device in a particular fashion. Other tools have also been developed to help control the rendering of content. Stylesheets is an example of one such tool. Stylesheets describe how a particular document (content) is to be presented on a screen, display, print, or otherwise. By attaching stylesheets to markup language structured documents (e.g. HTML) authors and readers can influence the presentation of the content without losing device independence.

The eXtensible Stylesheet Language (XSL) is a language for expressing stylesheets. One of the components of XSL is the XSL Transformations (XSLT) which is a language for transforming XML documents. A transformation expressed in XSLT is called a stylesheet because the transformation functions as a stylesheet - describes rules for transforming a source tree into a result tree. The stylesheet contains a set of template rules with each template rule having two parts: (1) a pattern which is matched against nodes in a source tree and (2) a template which can be utilized to form part of the result tree. The transformation is performed by associating patterns with templates. A pattern is matched against elements in the source tree. A template is instantiated to create part of the result tree. The result tree is separate from the source tree and, the structure of the result tree can be completely different from the

structure of the source tree. In constructing the result tree, elements from the source tree can be filtered and reordered, and arbitrary structure can be added. In operation, the stylesheet declares what output should be produced when a pattern in the XML document is matched. Additional information about the operation of the XSLT can be

5   obtained from the W3C or by visiting its website at the URL of http://www.w3.org.

Overall, the use of XSLT allows XML documents to be reformatted according to the parameters of XSL stylesheets, and presentation flexibility of the XML source can be obtained. This is an important function since it allows the separation of content from presentation - a key function for the extensibility and flexibility of the

10   world wide web. Chris Lilley, a W3C staff contact for the XSL Working Group states that with XSLT "we're closer to delivering rich, structured data content to a wider range of devices." Thus, XSLT allows a variety of devices that support varying markup languages to access content that is in a standardized format - XML.

Although XSLT helps to provide content to a variety of devices, several

15   problems are still present in the state of the art. One such problem is that the majority of web-based content is not formatted as XML documents but rather, as HTML documents. In addition, XML is not the easiest markup language to develop content with, thus, other, more user friendly languages are usually employed. Thus, to truly recognize the benefits of XSLT, there is a need for a mechanism to convert existing

20   content into XML. However, to convert a majority of the content that is available on the web into XML would be a daunting task. In addition, even if such a task could be performed, it would result in crippling the content provider community since the majority of content creators, as well as the tools used to create content, is more HTML oriented rather than XML. Thus, there is a need in the art for a mechanism to convert

25   content to XML, or some other interim format, when the content is being requested by a device. Such a mechanism would need to perform this task in a quick and efficient manner so as not to degrade the performance of accessing web content.

As previously stated, XML is a limited language that only uses a subset of the capabilities of SGML. Because of this, a transformation from one SGML format to

30   XML and then from XML to another SGML format may result in the loss of some

information. What is needed in the art is a mechanism to take content from one SGML format and translate it to another SGML format while minimizing the loss of any information. One technique to accomplish this would be to have a device or utility that directly transforms one SGML document into another SGML document.

5   Such a technique would be a tremendous improvement in the world wide web. Another technique to accomplish this would be to have a device or utility that transforms content from a variety of formats, including but not limited to SGML, into an interim format, and coupling this device or utility with another device or utility that would transform content from the interim format to a device specific format.

10   Internet content may be accessed from an ever increasing number and variety of devices. Today, such devices include desktop computers, laptop computers, hand held personal data assistants (PDA), cellular telephones, appliances, electronic kiosks, GPS devices and many more. In the future it is quite certain that additional devices will be used in the same manner. Each such device may have different display

15   capabilities and may utilize varying types of data interpreters. One device may use an SGML based web-browser, one device may use a WAP based browser and yet another device may use a non-markup language based interpreter. The above-described needs in the art are even more apparent when such a variety of devices are all attempting to access content from the world wide web. However, in addition to needing the

20   capability to transform content from one format to another, there is also a need in the art to determine what type of format and what additional format structuring is necessary in order to allow a device to render the content. When the transformations are being performed by a server-based environment rather than a client-based environment, the server must be able to identify the specific requirements of the

25   destination of the content, and interject these specific requirements into the transformation process.

        To illustrate the benefits that the present invention gives rise to, a need in the art regarding sever-based applications is provided. To utilize a server-based application, a client device connects to the server and initiates a download of the

30   application. The server acknowledges the request and initializes the application for

downloading (note that the application is also usually "compressed" for faster delivery over the Internet). The client device then downloads the compressed application "executable" from the server.

The application is now stored on the client, but is not yet ready for use. The application must first be unpacked or uncompressed, and then typically an installation routine that will make the application ready to run on the client must be invoked. Once this is done, the application can be invoked. Once invoked, the application runs resident in the client's memory. The application must be recalled from the hard drive of the client device, and executed each time it is desired to be used. In addition, the client likely has word processors, spreadsheets, and email editor programs, which must be recalled from its hard disk and made resident in its memory, to be executed. Each application consumes valuable resources of the client device. Thus, there exists a need in the art for a technique to minimize the expenditure of client resources when running server-based applications.

Another problem in the industry can be seen by examining software distribution techniques. Today, revisions or upgrades to application programs must be downloaded or shipped on a digital media ("shrink-wrapped") for installation by end-users. These upgraded versions are expensive to distribute and support costs are also significantly increased. End-users will incur costs of shipping, media and packaging to install the application, and significant problems may occur during these updates (incompatibility with existing software or systems installed on the end-user computer). Thus, there is a need in the art for a method to provide updated software without incurring the current expenses.

Therefore, it has been shown that there is a need in the art for devices that require content in a particular format, to access content through the Internet, or any other local or global network, regardless of the format in which the content is stored. In addition, there is a need in the art for a mechanism to convert content to XML, or some other interim format, when the content is being requested by a device. Such a mechanism would need to perform this task in a quick and efficient manner so as not to degrade the performance of accessing web content. There is also a need in the art

to determine what type of format and what additional format structuring in necessary in order to allow a device to render the content.

Finally, there exists a need in the art for a technique to minimize the expenditure of client resources when running server-based applications and for a method to provide
5   updated software without incurring the current expenses. The aforementioned needs in the art, as well as other needs are met by the various embodiments of the present invention that are described within this specification.

SUMMARY OF THE INVENTION

The present invention is directed towards solving the aforementioned needs in
10   the art, as well as other needs in the art, by providing a mechanism for transforming content between a variety of formats to allow the content to be rendered on devices that support differing content formats. One aspect of the present invention is that web-based content, which may be found in a variety of formats, can be rendered with a device that only supports a particular content format. Advantageously, this aspect of
15   the present invention extends the usefulness of the world wide web as a general content resource.

Another aspect of the present invention is a content transformer engine that includes a source content input, a memory element, and a conversion unit. The conversion unit receives source content in a first format at the source content input,
20   transforms the source content from the first format into an interim format; and stores the source content in the interim format in the memory element. In one embodiment of this aspect of the present invention, the content transformer engine may transform SGML formatted content. In another embodiment, the interim format may be XML.

Another aspect of the present invention is a content transformer engine that
25   includes a source content input, a destination content output, a destination specific parameter input and a conversion unit. The conversion unit receives source content in a source format at the source content input and at least one parameter from the destination specific parameter input. The conversion unit then transforms the source content from the source format into an interim format. The conversion unit also

identifies a destination format for the content, based at least in part on the destination specific parameter. The conversion unit then transforms the source content from the interim format to the destination format and provides the destination formatted content to the destination content output. In one embodiment of this aspect of the present invention, the content transformer operates on SGML content. The interim format may be XML or some other standard or proprietary format.

Another aspect of the present invention is a content transformer engine that includes a client identifier, a template database, a best fit analyzer, a linker database, an inbound content transformer and an outbound content transformer. The client identifier receives a content request from a client device and identifies characteristics about the client device from the content request, and by deriving additional information about the client device from implicit sources and previously stored information. The best fit analyzer selects a best fit template from the template database based, at least in part, on the client device characteristic. The linker database contains location information for content resources and access information for each of the content resources. The inbound content transformer receives source formatted content and transforms the source formatted content into interim formatted content. The outbound content transformer transforms the interim formatted content into destination formatted content in accordance with the best fit template. In one embodiment, the outbound transformer is an extensible style language transformation engine. In another embodiment, the inbound transformer is a reverse extensible style language transformation engine. In various embodiment, the characteristics of the client device may differ. Typical characteristics may include screen resolution, caching capabilities, device type, compression characteristics, decompression characteristics, network characteristics, bandwidth characteristics, and connection type. Thus, these embodiments and aspects of the present invention allow devices that require content in a particular format, to access content through the Internet, or any other local or global network, regardless of the format in which the content exists within the Internet. In addition, the present invention provides a mechanism to convert content to XML, or some other interim format, when the content is being

requested by a device performs this task in a quick and efficient manner so as not to degrade the performance of accessing web content.

In another embodiment, the present invention provides the ability for any SGML content to be requested over the Internet and displayed by any SGML-based

5   device making the request. Thus, this aspect of the invention allows the conversion of SGML content within the Internet and/or an intranet to be read and converted into a compatible format for any of a variety of receiving platforms. This exemplary embodiment of the present invention also reads information in an SGML format, converts the information to an interim format, and then renders the information into a

10   different or the same SGML format as originally read.

One utilization of the present invention, is in the delivery and remote access to server-based application programs. This utilization of the invention involves a client or remote computer system invoking an application program residing on a remotely located server system. The client is typically connected to the server system through a

15   network such as the Internet or an intranet. In operation, once a client invokes a server-based application program, the server system provides an application program interface to the client. The application interface can easily be adapted for compatibility with a variety of platform types. The server system obtains platform information from the client, and then constructs and provides a compatible application

20   program interface to the client.

These aspects and embodiments, as well as other aspects and embodiments of the present invention are more clearly described in the specification, figures and claims.

BRIEF DESCRIPTION OF THE DRAWINGS

25   Fig. 1a is a block diagram illustrating the functional components of an exemplary embodiment of the transformation engine.

Fig. 1b is a block diagram illustrating the functional components of an exemplary embodiment of the transformation engine that utilizes an interim content format.

Fig. 2 is a system diagram illustrating an exemplary environment in which the transformation engine can be utilized as a server-based content transformer.

Fig. 3 is a block diagram illustrating further details of the server-based content transformer described in Fig. 2.

5   Fig. 4 is a process flow diagram illustrating a particular application of the present invention in providing a device-agnostic interface capability for server-based applications.

DETAILED DESCRIPTION

The present invention provides a mechanism for transforming content between

10   a variety of formats through the use of a transformation engine. In a general embodiment of the present invention, the transformation engine reads content in one format and transforms the content into another or the same format. Turning now to the figures in which like numerals, letters and labels refer to like elements throughout the figures, various embodiments and applications of the present invention are

15   provided.

Fig. 1a is a block diagram illustrating the functional components of an exemplary embodiment of the transformation engine. The transformation engine **100a** operates to transform source content **110** from its existing format to a destination content format **130** that is compatible with a destination target. The transformation

20   engine **100a** includes an s to d content transformer **140**, where s and d are variables identifying the format for the source content and the destination content respectively. In operation, the content transformer **140** receives or extracts content stored in a source format (content$_s$) **110** and transforms the content$_s$ into a destination format (content$_d$) **130**. This transformation process may include the incorporation of

25   destination specific parameters **135** to control the operation of the transformation. The transformation engine **100a** provides a bi-directional transformation of content, meaning that content can be transformed from and into any of a variety of formats. This aspect of the present invention enables universal transformation of content from any type to any type, rapidly and at large scale and volume. This capability is not

presently available for sharing of content via the Internet, and intranet or other global or local networks.

Various embodiments of transformer engine **100a** may be used to perform specific content translations. In a particular embodiment, the transformer engine **100a** may be used to transform content between SGML formats. In this embodiment of the transformer engine **100a**, source SGML **110** is input into the content transformer **140**. The content transformer **140** then renders the source SGML into a destination SGML format **130**. In this embodiment of the transformer engine **100a**, the functionality of the content transformer **140** is provided through the use of a modified XSLT engine.

XSLT is an open standard that is sponsored and supported by the W3C organization. Those skilled in the art will be familiar with the operation of XSLT; however, additional information may be obtained from the URL http://www.w3.org/Style/XSL/. The standard XSLT engine takes any XML content and renders it into SGML content or another variant of XML. The XSLT engine is a rules based conversion that examines the content on a tag level and converts the tags and groups of tags into an equivalent or near equivalent set of tags for the desired format. A simple example can be illustrated in the operation of the standard XSLT engine with a break tag. In XML format, a break is portrayed as <br/>. Invoking the XLST engine with HTML identified as the destination format would result in converting <br/> into <br>, the format used by HTML. The standard XSLT engine includes rules for transforming XML to a larger variety of other SGML formats. XSLT is based on template rules which specify how XML documents should be processed. Although conventional programming languages are often sequential, template rules can be based in any order because XSLT is a declarative language. A stylesheet declares what output should be produced when a pattern in the XML document is matched.

The modified XSLT engine of the present invention expands the capabilities of the standard XSLT engine by allowing the ability to identify a destination format, as well as a source format. The standard XSLT engine already includes the rules and stylesheets for multiple SGML formats. This aspect of the present invention expands

12

the use of these rules and stylesheets to all the transformation of one SGML format to another. The determinative factors for the destination SGML format include input parameters and information about the destination for the content, illustrated as destination specific parameters **135** in Fig. 1a.

5   Fig. 1b is a block diagram illustrating the functional components of an exemplary embodiment of the transformation engine that utilizes an interim content format. The transformation engine **100b** operates to transform source content **110** from one its existing format to a format destination content **130** that is compatible for a destination target. The transformation engine **100b** includes two main components:

10   (a) an inbound content transformer **115** (source format to interim format) and (b) an outbound content transformer **125** (interim format to destination format). In operation, the inbound content transformer **115** receives or extracts content stored in a source format (content$_s$) **110** and transforms the content into an interim format (content$_i$) **120**. Content$_i$ **120** may be stored for future use or it may immediately be

15   processed by the outbound transformer **125** and then discarded. Advantageously, this aspect of this embodiment of the present invention allows content to be staged or temporarily stored in the interim format. If the client device, or even a server device in a server-to-server content exchange, is too slow to receive the content as it is transformed, the interim format is ideal for staging the content until the destination is

20   ready to receive it.

The outbound content transformer **125** transforms the content$_i$ **120** into a destination format (content$_d$) **130**. This outbound transformation process may include the incorporation of destination specific parameters **135** to control the operation of the outbound transformation. The transformation engine **100b** provides a bi-directional

25   transformation of content, meaning that content can be transformed into a standard or proprietary interim format from a device specific format, and from the standard or proprietary interim format into another or the same device specific format. This aspect of the present invention enables universal transformation of content from any type to any type, rapidly and at large scale and volume. This capability is not

presently available for sharing of content via the Internet, and intranet or other global or local networks.

Various embodiments of transformer engine **100b** may be used to perform specific content translations. In a particular embodiment, the transformer engine **100b** may be used to transform content between SGML formats. In this embodiment of the transformer engine **100b**, source SGML **110** is the input to the inbound content transformer **115**. The inbound content transformer **115** then renders the source SGML into an interim format **120**. Preferably the interim format **120** is an industry standard format, such as XML, however any industry standard or proprietary format that allows flexibility can be used. The transformed content in the interim format **120** may be stored in memory or a storage repository for further processing by the outbound content transformer **125**. The content is then transformed from the interim format **120** by the outbound transformer **125** into the destination SGML format **130**.

In this embodiment of the transformer engine **100b**, the outbound transformer **125** function could be performed by employing the use of an XSLT engine. The inbound content transformer **115** is basically a reverse operation of the standard XSLT engine. Rather than utilizing the rules of XSLT to transform from XML to SGML, in reverse operation, the XSLT engine transforms SGML to XML. In this manner, the transformer engine **100b** may convert any source SGML **110** content to any other determinative destination SGML **130** content.

Therefore, in this embodiment, the modifications to the XSLT engine enable the bi-directional transformation of content. Standard XSLT engines, such as Sablotron developed by Ginger Alliance or Unicorn XSLT Processor developed by Unicorn Enterprises, only allow "XML to SGML" transformations. This embodiment of the transformer engine **100** utilizes a modified XSLT engine to enable a reverse transformation "SGML to XML". Thus, this embodiment of the present invention enables universal transformation of content from and to any SGML format, rapidly and at large scale and volume.

It should be clear to those skilled in the art that the use of the XSLT engine and XML is simply one embodiment of the present invention that may be chosen due

14

to convenience, standardization and availability of compatible content and tools. Other transformer engines that convert between other formats may also be employed and the present invention anticipates the development and the use of such other transformer engines.

5   Fig. 2 is a system diagram illustrating an exemplary environment in which a transformation engine **100** can be utilized as a server-based content transformer. In this environment, the transformer engine **100** is incorporated into an Internet connected server **200**. This embodiment of the present invention is referred to as the NIMBUS server engine by the developers of Melia Design Group but, is simply

10   referred to as a server engine **200** herein. Prior to addressing the details of the server engine **200**, the general operation and environment for the server engine **200** is provided. The server engine **200** may be used to convert web based or Internet-based content to any of a variety of platforms. For illustrative purposes, Fig. 2 provides a exemplary platforms that may employ the use of the server engine **200** (pen computer

15   **210**, PDA **220**, portable computer **230**, personal computer **240**, interactive TV **250** through broadcasting station **255**, cellular telephone **260** through mobile telephone switching office **265**, and other sources of content **270**). The server engine **200**, along with the exemplary platforms, is communicatively coupled to the Internet **290**, or some other global or local network. Devices wishing to access content may do so

20   through the server engine **200**. In operation, the server engine **200** receives a request for content from a device. In response to such a request, the server engine **200** obtains the content for the device as well as information about the device. The server engine **200** then transforms the content from it present format, to a format that is compatible with the requesting device.

25   As an illustrative example, suppose that a user of cellular telephone **260** desires to obtain a stock quote. Assuming the cellular telephone **260** is equipped with a hand held device markup language browser, the user may browse through a list of bookmarks and select a specific bookmark that will request the stock quote. The server engine **200** supporting the cellular telephone **260** will receive the request.

30   Depending on the particular implementation, the server engine **200** may have pre-

stored information about the cellular telephone **260** or, the server engine **200** may enter a process to obtain the information about the cellular telephone **260**. In addition, the server engine **200** performs the necessary functions or operations to obtain the stock quote. The stock quote may be in a variety of formats including, HTML or

5   simply ASCII text provided through an online feed. Regardless of the format or source of the content, the server engine **200** extracts the requested content and converts it to a format that is compatible with the cellular telephone **260**, which in this case would be HDML. The server engine **200** then provides the requested content to the cellular telephone **260** which then renders the information on its display device

10   **261**.

From this illustrative example, it should be evident to the reader that the present invention provides tremendous benefit in the access and sharing of information within a global or local network. Each of the devices illustrated in Fig. 2 may be a content source and a content destination. In addition, each device may

15   require a different format for the content. However, the present invention may be utilized to enable each of the devices to share and receive content from any of the other devices.

Fig. 3 is a block diagram illustrating further details of the operation of the server-based content transformer described in Fig. 2. The overall operation of this

20   embodiment of the present invention can be described in five general processes: (1) Client Identification, Determination and Parsing; (2) Content Storage, Indexing and Retrieval; (3) Content Transformation; (4) Content Packaging; and (5) Content Delivery.

Client Identification, Determination and Parsing. In this embodiment of the

25   present invention, a client device **305** initiates the acquisition of content by issuing a content request **307** to the server engine **200**. In response to receiving the content request **307**, the server engine **200** attempts to identify the characteristics or device specific information regarding the client device **305** and/or the network from which the client device **305** is operating. Depending on the particular content and structure of

30   the content request **307**, the server engine **200** may have sufficient information to

proceed, or may require additional information from the client device **305**. If additional information is required, the server engine **200** may invoke a series of handshaking or communication exchanges with the client device **305**. In either case, the server engine **200** must obtain device specific or environmental variables regarding

5   the client device **305**. The client device **305** illustrated in Fig. 3 shows three categories of environmental variables. The categories and specific environmental variables provided are intended to be illustrative rather than exhaustive and different devices may include or require different categories or variables. The server engine **200** may determine the client variables from a variety of sources or techniques. A

10   primary source for determining the variables in a web-based request is the HTTP request header since the content request from the client device **305** is normally embedded within a URL string. The server engine **200** may also utilize the "round trip" time and other implied or explicit information to determine the value of the variables.

15   The variables identify specific content that the client is requesting. Some of the client variables identify the type of device making the request, client screen resolution, the client caching capabilities (on client device and on client network), the device display specifics (resolution, color/B&W/greyscale, type, color bit depth, etc.), and network characteristics (bandwidth, bandwidth/connection type or protocol,

20   variability of bandwidth, compression/decompression rates, etc.). Those skilled in the art will realize this list is not exhaustive and that in any given request each of these variables, fewer than those listed, or more than those listed may be used.

Once the variables and their values have been identified, the server engine **200** performs request parsing **310**. One aspect of the request parsing process **310** is to

25   perform a "best fit" analysis by evaluating the variables against an existing "best fit matrix" in order to derive a content category. The best fit matrix is simply a single to multidimensional matrix that receives one or more variables as an input and provides a "best fit" template or profile based on the variables. In some instances, only a single variable may be required. For instance, if the server engine **200** has received a

30   variable that identifies the client device **305** as a specific version and model of a Palm

IV PDA, this variable may be used as a single lookup variable to identify the best fit profile. However, in other instances, multiple variables may be required to identify a best fit template. For instance, to identify a best fit template for a hand held wireless device, the server engine **200** may require the following variables: a device type, a

5   display type, a display resolution, and a service grade level.

The content category is utilized by the server engine **200** in an effort to determine a "best fit" method or profile for: (a) sending the correct image, video, text, template, and markup language (HTML, DHTML, SGML, XHTML, XML, etc.) that optimizes display and performance on the client device **305**; (b) sending the correct

10   content density for the client's available/effective bandwidth; (c) determining which methodology is appropriate for enabling transactions; and (d) determining what session variables, etc. are available, and which variables are necessary for negotiating and employing appropriate security, encryption algorithms, etc. If the system is unable to determine a "best fit" profile from existing content categories and the client

15   variables, then a best fit value (lowest common denominator fit) is determined and used until enough information related to each of the above variables can be collected.

These client variables are then evaluated against a user's profile **315** (derived from previous interactions with the server engine **200**, and supplemented with any new or changed information from the current client identification and determination

20   process). The server engine **200** can be extended to enable the use of shared profiles or parameters (for instance, all AOL users may have similar characteristics, so there is no need to fully evaluate each AOL user independently). The server engine **200** may also be extended for adaptive learning. Using adaptive learning, as user profiles are updated, corrected, or new information is obtained during the user's interaction with

25   the server engine **200**, the user's profile is updated to reflect these profiles, corrections and new information.

Thus, during the client identification, determination and parsing process, the server engine **200** extracts information from a content request **307** from a client device **305**, utilizes existing information about the client device **305** and/or a user of the

30   client device, and derives information from other sources in an attempt to identify (a)

18

what content to obtain for the client device **305** in response to the content request **307** and (b) how to deliver the content to the client device **305**.

Content Storage and Retrieval. The server engine **200** maintains a meta-database **320** (also referred to as a linker database) that contains information about the

5   content resources **330** that are indexed by the server engine **200**. A content resource **330** may be any of a variety of devices or entities that provide content or information including, but not limited to, remotely located systems accessible via the Internet or some other means, local systems, live audio or video feeds, live text feeds such as news sources or the like and databases.

10   Typically, the linker database **320** does not store the actual content, but rather stores information about how (resource retrieval API **322**) and where (content resource location **324**) to obtain the content. Advantageously, the use of a linker database **320** allows the number and amount of content resources **330** available to the server engine **200** to be virtually limitless.

15   More specifically, for each content resource **330**, the linker database **320** maintains descriptive information to facilitate the server engine's **200** access of content from the content resource **330**. Any of a variety of techniques may be utilized to perform this function. In one embodiment, for each particular content resource **330**, the linker database **320** maintains (a) a unique system identifier, (b) a resource

20   name, (c) a template required for the display of the available content (a sub-table with various template types), (d) the location and state or status of the available content, (e) the type of content (image, video, streaming video, text, markup language document, application component, etc.), (f) the format of the available content (.jpeg, .mpeg. etc.), (g) the version of the content format for the available content (MPEG2, MPEG3,

25   etc.), (h) the content quality of the available content (color bit depth, resolution, frame size, frames per second, etc.), (i) instruction sets or business rules that describe use information regarding the available content (last use, last used by, date added, date updated, copyrighted or public domain, etc.), (j) a cross-reference to another repository that may contain information necessary to retrieve and utilize the content

30   (i.e., structure query language statement, application program interface, lookup

statement, etc.) and (k) other information that may be necessary to optimize the performance of the server engine **200** in the access, conversion and delivery of the content such as, the speed and throughput of the network connection between the server engine **200** and the content resource **330**, and the amount of time required to

5   render the content.

Thus, during the content storage and retrieval process, the server engine **200** utilizes the information obtained during the client identification, determination and parsing process, as well as information contained in the linker database **320** to obtain the content requested in the content request **307**, as well as additional information

10   regarding the content. Based on the client variables, the server engine **200** may seek out and obtain content that is optimally suited for the client device **305**. Advantageously, this aspect of the present invention helps to minimize processing that is required in the transformation of the content. The content, along with any other necessary information is then provided to the inbound content transformer **115**.

15   Transformation, Packaging and Delivery Process. During the transformation, packaging and delivery process, the server engine **200** transforms the content to a format suitable for the client device **305**, and then packages and delivers the transformed content to the client device **305**.

The content identified in the content request **307** and retrieved from one or

20   more content resources **330** is, based on the "best fit" content category, transformed to a format that is suitable for the client device **305**. In the embodiments described in conjunction with Fig. 1, this process may be accomplished by transforming the source content to an interim format **120** and then based on the "best fit" template transforming the content into a destination format **130**, or the source content can be

25   directly transformed from the source format **110** to the destination format **130** based on the "best fit" template.

In a specific embodiment of the server engine **200**, the outbound content transformer **125** may be an XSLT engine and the inbound content transformer **115** may be a reverse XSLT engine. In this embodiment, the source content will be

30   transformed from a source format (SGML$_s$) **110** to an interim format (XML) **120** and

20

then to a destination format (SGML$_d$) **130**. The server engine **200** uses business rules and the "best fit" content category derived during the client identification, determination and parsing process to select the appropriate SGML format for the client device **305**. Other engines and engine variants may also be used or developed

5   and the present invention anticipates such development.

Once the content has been transformed, the server engine **200** "wraps" or "overlays" the content with an appropriate template. Once again, based on the "best fit" analysis and the "best fit" content category, the server engine **200** determines a proper set of templates and template parameters for displaying the content and merges

10   the transformed content into the proper template for delivery to the client device **305**. The linker database **320**, in addition to indexing content resources **330**, also indexes a template database **340**. The template database **340** may be a local database or a remotely located database accessible to the server engine **200** through the Internet or some other localized or global network. Based on the "best fit" analysis, the location

15   **326** of the "best fit" template can be extracted from the linker database **320** and used to access the "best fit" template from the template database **340**.

Thus, the server engine **200** will provide content to the client device **305** in a format that is adapted to the client device **305**. The adaptation to the client device **305** is a function of the information indexed from the content resources **330**, but is also a

20   function of the template chosen to render the information to the client device **305**. This template will be indexed from the template database **340** and its content will be dependent on the values of the client variables and the variables of the content request **307**. The template database **340** is a second functional input to the transformation engine **100** and thus, controls the way the output is rendered to the client device **305**.

25   The output **130** of the server engine **200** is provided to the client device **305** through a network connection, such as through the Internet. Advantageously, the present invention operates to obtain, transform and deliver the compatible content in a quick and efficient manner.

This aspect of the present invention further enables the dynamic

30   transformation of even currently incompatible content types for a given display

device. For instance, content that was formerly too large (in file size, display format) for a Personal Digital Assistant (PDA) can be "thinned down" based on a "best fit" basis to make the content usable to the PDA. Content transformation can occur against content that is stored within the server engine, or against content that is

5   remotely indexed -- stored on servers and in repositories (databases, file systems, etc.) that are not proximate to, or related to the server engine. This remote indexing of content very closely follows "peer-to-peer" content indexing (e.g. music sharing systems), but is different in that the content may be of many disparate and heterogeneous types and formats (pictures, video, music files, stock ticker data, text

10   documents, HTML, etc.); thus more akin to a server-to-server functionality.

It should also be noted that in alternative embodiments, the server engine **200** may offer integrated security. Integrated security includes application components, template design, database access, and application access, all determined by the same security (users & groups) level of access. This keeps all access to the server engine

15   **200** under a single governing system. The server engine **200** circumvents standard browser based security messages by intercepting these messages, processing them (as normal content) into the server engine **200**, "wrapping" the security dialog in an appropriate template, and rendering them in a consistent format as a "containerized application" through which the user can interact within the System.

20   **Applications of the Present Invention**

The present invention, as described above, is revolutionary in its ability to access content in any format and deliver the content in a compatible format to another device. In addition, there are many revolutionary applications of the present invention that, prior to such invention, were not available or possible to implement. Two such

25   applications include (1) a content manager and (2) platform agnostic server-based applications.

Content Manager. Within the context of a content manager, the present invention empowers users at all skill levels, whether they have little to no HTML or other markup language knowledge or experience, to create and manage content. The

30   transformer engine aspect of the present invention allows the content of a web site to

be stored in an interim format and then accessed and transformed for any of a variety of platforms. In addition, as will become more apparent upon reading the description of the platform-agnostic server-based applications, the web site content can be easily edited and managed, from any platform, using a user friendly, intuitive interface.

5   The present invention can be utilized to create different toolsets. The toolsets are functional aspects that enable the actuation of content (e.g. print function, cut function, paste function, bold, italics, underline, size, symbols, justify, bullets, number, indent, color, linking, etc.). Though initial applications have been developed using the toolset of the present invention, those skilled in the art will realize that other

10   applications and other toolsets that leverage the present invention can be used/reused to create more applications and toolsets. Thus, numerous embodiments of this invention include different toolsets and application combinations.

The content manager based on the present invention provides a "create once, publish anywhere" (COPA) capability. Thus, content can be created and then

15   rendered in any format to any device. The ability to populate a system once for multiple devices is very advantageous. The content managers can handle a number of different markup language, including but not limited to, wireless, handheld devices and set-top boxes, in addition to the variety of formats on the Web. The content manager allows a user to preview the content on multiple devices before the content is

20   published. The content manager also allows a user to create different sets of device templates.

Platform-Agnostic Server-Based Applications. Fig. 4 is a process flow diagram illustrating a particular application of the present invention in providing a platform- agnostic interface capability for server-based applications. In general, the

25   present invention provides for the capability for server-based applications to be shared with a variety of platforms. User interface information is stored in an interim format on the server and then rendered to device specific formats when the applications are invoked or actuated by a client device. Thus, the server-based applications can be shared with any of a variety of platforms.

Initially, the client device **305** invokes a server-based application by issuing a content request **405**. The server-based application may reside on the server engine **200** or any remote server accessible by the server engine **200**. In response to receiving the content request **405**, the server engine **200** identifies the specifics of content

5   request, invokes the application and produces a client specific application interface **410**. The server engine **200** then delivers content defining the application interface **415** to the client device **305**.

.   This application interface provided to the client device **305** may be a custom application interface that runs resident on the client device **305**, a browser-based

10   interface that runs resident within a browser application running on the client device **305**, or a browser-like application interface which runs resident on the client device **305**. For a personal computer user, a browser-based application may have the look and function of an "executable" compiled 32-bit processing calculator application interface running on the clients computer. However, in reality, only a shell interface is

15   provided to the client and a portion of the processing is performed on the server. This invention is not limited, however, solely to 32-bit interfaces, but can be automatically adapted to provide an interface that is similar to the operating system and programming style of the end user's client.

Upon receiving the content defining the application interface **415**, the client

20   device **305** renders the application interface on its display and then waits for a user or process to actuate a feature of the application interface. The performance of, and actuation of, the server-based application will be exactly like, or similar to, the actuation required for a compiled, executable application. Once the application interface is actuated **420**, the actuation will either be handled locally by the client

25   device **305** or remote to the client device **305**. For local processing **425**, the client device **305** simply interprets the actuation and performs the appropriate action. As an example, if the server-based application is a calculator, actuating a digit key may be processed locally and thus, the client device **305** would render a number on the display. If the actuation is to be processed remotely **425**, the client device **305**

30   provides a content request **430** to the server engine **200**.

In response to receiving the content request **430**, the server engine **200** identifies specifics of the content request, invokes the appropriate action, and produces a modified application interface if necessary **435**. Once the modified application interface is produced, the server engine **200** provides the content **440** to

5   the client device **305**. Additional actuation may be performed by the client device **305** and each will be processed as described above.

Eventually, the client device **305** may attempt to exit the application **440**. If an exit with a request to save any modifications is performed **445**, the client device **305** provides an exit request and any data that needs to be saved **450** to the server engine

10   **200**. In response to receiving the exit request with a save **450**, the server engine **200** updates any necessary databases with the changes **455**. If an exit without a request to save any modifications is performed **445**, the client device **305** provides and exit request **460** to the server engine **200**. Whether the exit request is with or without a save operation, the server engine **200** then closes or terminates the application **465**.

15   The server engine's distinct ability to remotely index any content, also applies to these applications of the present invention. For the content manager, components (source code modules to support menu drop downs, buttons, dialog boxes, pop-up windows, subroutines, etc.) can be remotely index, and dynamically served.

The content manager application is actually a specific embodiment of the

20   device-agnostic server-based application implementation of the present invention. The content manager enables a client device to access and change Internet delivered (website) content from an application that is provided to the client device from a remote server, using code components and controls that are only delivered via an Internet browser or browser-based application interface or some other application

25   interface.

This application of the present invention illustrates the impact of the present invention in Internet and intranet publishing and content management fields, as well as the field of software and hardware development related to the delivery to remote telecommunications devices such as pagers, Personal Digital Assistants ("PDAs"),

30   interactive TV, digital set-top boxes, and the like. Application of this invention

25

enables persons engaged in the creation or maintenance and delivery of content (websites, programs, interactive television, data, etc.) to use remote-server-based software components to alter the content without requiring:

(a) The downloading, installation and execution of compiled code components
5   on the user's "client";

(b) A "browser add-on application or applet" to be created or executed from the local (client) machine or server; and/or

(c) HTML, DHTML, XML, Active Server Pages, or other Internet Markup language expertise or experience.

10 Advantageously, this application of the present invention reduces the requirement for the physical distribution of software, requiring the user to purchase media and documentation that includes the cost of that distribution. There is no need to download and install the application, and immediate upgrades are made available as a result of the invention's functionality.

15 Another feature of the system is the ability to maintain the state of the application in memory or storage of the server, enabling the end user to suspend interaction with the application for a period of time from their client, and later, to pick up "where they left off" using the same client, or a different client. For instance, the end user may perform some calculations using his or her PDA, to stop those
20 calculations for a period of time, and later, to begin again at the same point in the calculative process from the personal computer in the end user's office. Regardless of device or interface, the functionality will be similar, and the information and processing provided by the calculator will be the same, regardless of the end-user's display device.

25 **Conclusion**

It should be apparent to the reader that the present invention is unique in the industry and provides a revolutionary ability for sharing applications via the Internet.

The present invention has been shown to include a transformation engine that enables content and information to be transformed from one format, a source format,
30 to a format that is compatible with the requesting device, a destination format. This

aspect of the present invention allows for varying devices to access and share content via a network without concern as to the format of the content. In addition, the present invention enables applications that were not available prior to this invention. The examples herein have been provided for illustrative purposes only and should in not

5   be interpreted as restricting any aspects of the present invention. Alternate embodiments will become apparent to those skilled in the art to which the present invention pertains without departing from its spirit and scope. Accordingly, the scope of the present invention is described by the appended claims and supported by the foregoing description.

CLAIMS

What is claimed is:

1.   A content transformer engine comprising:

a source content input;

5   a memory element;

a conversion unit, the conversion unit being operative to:

receive source content in a first format at the source content input;

identify the first format of the source content and transform the

10   source content from the first format into an interim format; and

store the source content in the interim format in the memory element.

2.   The content transformer engine of claim 1, wherein the first format is one of a plurality of SGML formats.

15   3.   The content transformer engine of claim 1, wherein the interim format is XML.

4.   The content transformer engine of claim 1, wherein the first format is one of a plurality of SGML formats and the interim format is XML.

5.   The content transformer engine of claim 1, further comprising:

20   a destination content output;

a destination specific parameter input;

and the conversion unit is further operative to:

receive at least one parameter from the destination specific parameter input;

25   transform the source content from the interim format into a second format, the second format being selected from a plurality of potential formats based at least in part on the at least one parameter; and

provide the source content in the second format to the destination content output.

6.   The content transformer engine of claim 5, wherein the first format is one of a plurality of SGML formats.

7.   The content transformer engine of claim 5, wherein the interim format is XML.

8.   The content transformer engine of claim 5, wherein the first format and the second format are one of a plurality of SGML formats and the interim format is XML.

9.   A content transformer engine comprising:

a source content input;

a destination content output;

a destination specific parameter input; and

a conversion unit, the conversion unit being operative to:

receive source content in a source format at the source content input;

receive at least one parameter from the destination specific parameter input;

transform the source content from the source format into an interim format;

transform the source content from the interim format to a destination format, the destination format being selected from a plurality of potential destination formats based at least in part on the at least one parameter; and

provide the destination formatted content to the destination content output.

10.   The content transformer engine of claim 9, wherein the source format is one of a plurality of SGML formats.

11.   The content transformer engine of claim 9, wherein the interim format is XML.

12.   The content transformer engine of claim 9, wherein the source format and the destination format are one of a plurality of SGML formats and the interim format is XML.

13.   An SGML content transformer engine comprising:

an SGML data input;

an SGML data output; and

a conversion unit, the conversion unit being operative to:

5   receive input data in a first SGML format at the SGML data

input;

convert the input data into an interim format;

convert the input data from the interim format to output data,

the output data being in a second SGML format based on the characteristics of an

10   intended target for the output data; and

provide the output data to the SGML data output.

14.   A content transformer engine comprising:

a source content input;

a memory element;

15   a conversion unit, the conversion unit being operative to:

receive source content in a first format at the source content

input, the first format being one of a plurality of SGML formats;

identify the first format of the source content and transform the

source content from the first format into a second format, the second format being one

20   of a plurality of SGML formats; and

store the source content in the second format in the memory

element.

15.   A content transformer engine comprising:

a source content input;

25   a destination content output;

a destination specific parameter input; and

a conversion unit, the conversion unit being operative to:

receive source content in a source format at the source content

input, the source format being one of a plurality of SGML formats;

receive at least one parameter from the destination specific parameter input;

transform the source content from the source format to a destination format, the destination format being selected from a plurality of potential

5   destination formats based at least in part on the at least one parameter; and

provide the destination formatted content to the destination content output.

16.   The content transformer engine of claim 15, wherein the interim format is XML.

10   17.   The content transformer engine of claim 15, wherein the destination format is one of a plurality of SGML formats.

18.   An SGML content transformer engine comprising:

an SGML data input;

an SGML data output; and

15   a conversion unit, the conversion unit being operative to:

receive input data in a first SGML format at the SGML data input;

convert the input data from the first SGML format to output data, the output data being in a second SGML format based on the characteristics of

20   an intended target for the output data; and

provide the output data to the SGML data output.

19.   A content transformer engine comprising:

a client identifier, the client identifier being operative to receive a content request from a client device, identify client device characteristics from the

25   content request, and by deriving additional information about the client device from implicit sources and previously stored information;

a template database;

a best fit analyzer, the best fit analyzer being operative to select a best fit template from the template database, the best fit template is being selected, based at

30   least in part, on the client device characteristics;

a linker database, the linker database containing location information for a plurality of content resources and access information for each of the plurality of content resources;

an inbound content transformer, the inbound content transformer being operative to receive source formatted content and transform the source formatted content into interim formatted content; and

an outbound content transformer, the outbound content transformer being operative to transform the interim formatted content into destination formatted content in accordance with the best fit template.

20. The content transformer engine of claim 19, wherein the outbound transformer is an extensible style language transformation engine.

21. The content transformer engine of claim 19, wherein the inbound transformer is a reverse extensible style language transformation engine.

22. The content transformer engine of claim 19, wherein characteristics of the client device comprise at least one of the following characteristics:

screen resolution;

caching capabilities;

device type;

compression characteristics;

decompression characteristics;

network characteristics;

bandwidth characteristics; and

connection type.

23. A method for providing access to network-based content, the content existing in one of a plurality of SGML formats, to any of a plurality of devices, each of the plurality of devices supporting one of the plurality of SGML formats, the method comprising the steps of:

receiving a content request from a particular device, the content request identifying at least a desired content;

obtaining characteristic parameters about said particular device;

identifying a content source for the desired content, the desired content residing in a first SGML format;

converting the desired content to an interim format;

converting the desired content from the interim format into a particular

5   SGML format based on the characteristic parameters, the particular SGML format being compatible with the particular device; and

delivering the desired content in the particular SGML format to the particular device.

24.   The method of claim 23, wherein the interim format is XML.

10   25.   A method for providing access to network-based content, the content existing in one of a plurality of SGML formats, to any of a plurality of devices, each of the plurality of devices supporting one of the plurality of SGML formats, the method comprising the steps of:

receiving a content request from a particular device, the content request

15   identifying at least a desired content;

obtaining characteristic parameters about said particular device;

identifying a content source for the desired content, the desired content residing in one of a plurality of SGML formats;

converting the desired content into the XML format;

20   converting the desired content from the XML format into a particular SGML format based on the characteristic parameters, the particular SGML format being compatible with the particular device; and

delivering the requested content to the particular device.

26.   A method for providing access to information stored in one of a

25   plurality of SGML formats to a device supporting one of a plurality of SGML formats, the method comprising the steps of:

receiving a content request from a particular device, the content request identifying at least a desired content;

obtaining characteristic parameters about said particular device, said characteristic parameters at least identifying a desired SGML format and a display resolution for said particular device;

identifying a content source for the desired content, the desired content
5   residing in one of a plurality of SGML formats;

converting the desired content into an interim format;

converting the desired content from the interim format into the desired SGML format based on the characteristic parameters; and

delivering the requested content to the particular device.

10   27.   A server engine for obtaining and delivering desired content to a client device, the desired content residing in a first SGML-based format on a content resource, wherein said desired content may be converted to a second SGML-based format prior to delivery to said client device, said server engine comprising:

a client device interface for receiving a content request from a client
15   device;

a client identifier that is operative to interpret the content request to identify characteristics of the client device, interpret the characteristics of the client device to identify the second SGML-based format, and the desired content;

a linker database for identifying a content resource containing the
20   desired content; and

a transformer engine operative to transform the desired content from the first SGML-based format into the second SGML-based format.

28.   The server engine of claim 27, wherein said server engine includes a
25   template database, further comprising:

a best fit analyzer operative for identifying a best fit template by retrieving constants stored in said template database which correlate to the characteristics of the client device and incorporating the constants into the best fit template; and

the transformer engine is further operative to transform the desired content based on the best fit template.

29.   The server engine of claim 27 wherein said server engine includes an extensible style language transformation engine.
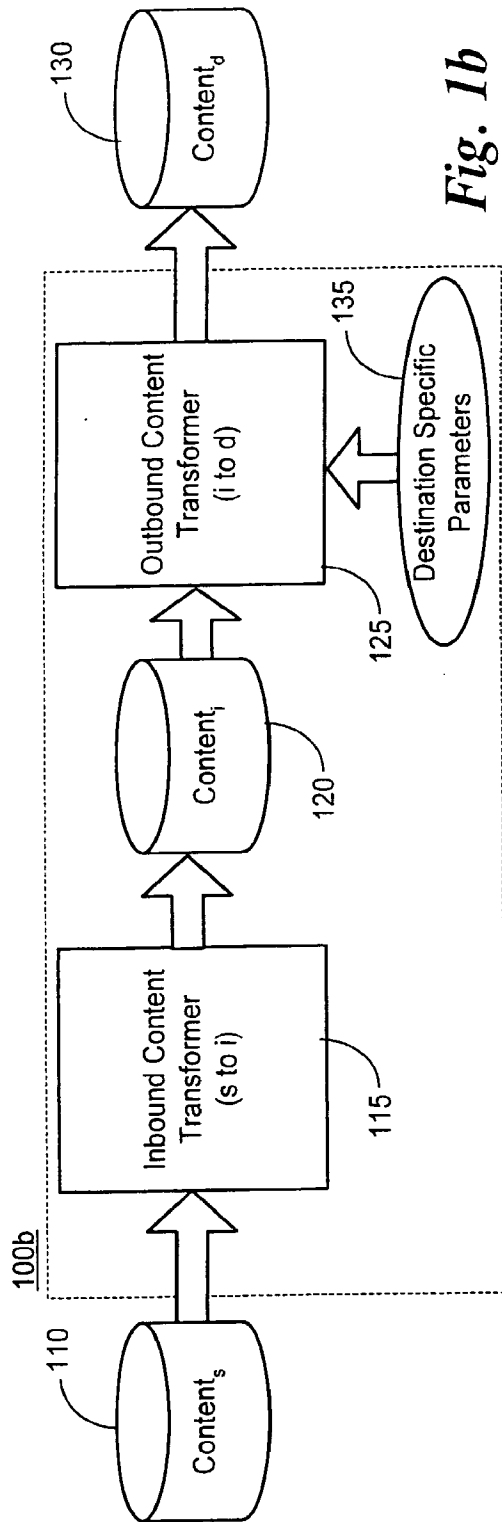
30.   The server engine of claim 27, wherein the characteristics of the client device comprise at least one of the following characteristics:

screen resolution;

caching capabilities;

device type;

compression characteristics;

decompression characteristics;

network characteristics;
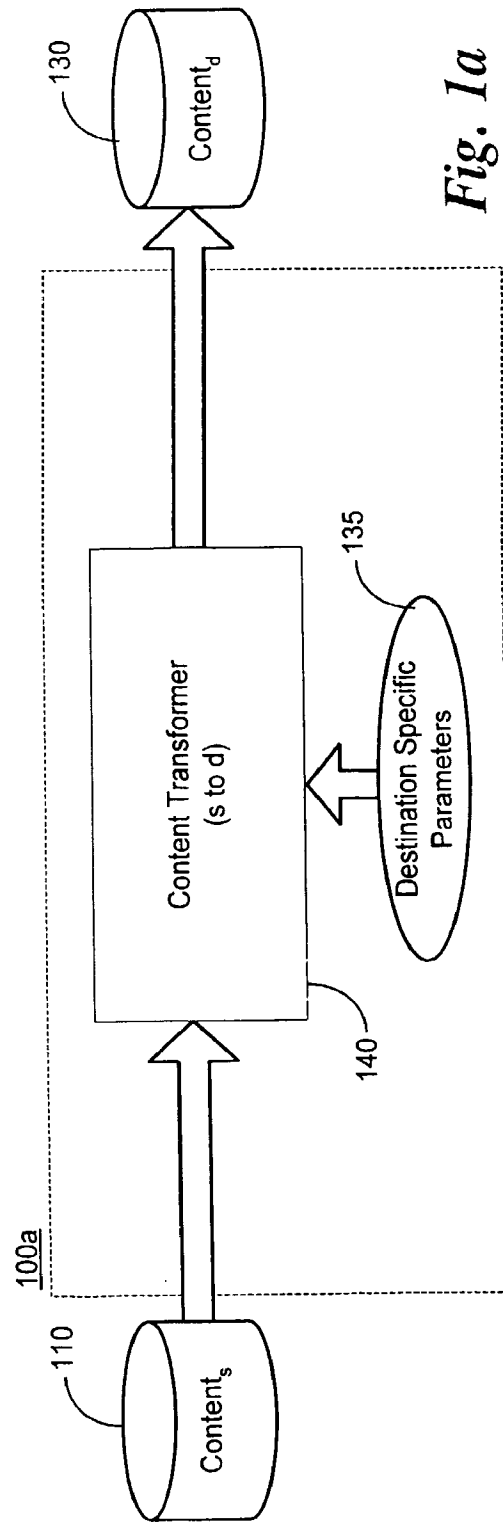
bandwidth characteristics; and

connection type.

31.   A method for allowing various platforms to execute server-based applications through a network, the method comprising the steps of:

receiving an invocation request from a client device, the invocation request identifying the client device and a server-based application;

selecting an application interface particular to the client device;

providing rendering information for the selected application interface to the client device;

invoking the server-based application;

receiving an action request from the client device;

interpreting the action request to identify changes required to the application interface;

providing rendering information for the changes to the application interface to the client device; and

invoking a server-based application command in accordance with the action request.

32.   A system for remotely accessing a server-based application, the system comprising a client and a server communicatively coupled to the client through a network, the client being operative to:

invoke the server-based application by sending an invocation request to
5   the server;

receive an SGML-based application interface;

display said SGML-based application interface;

receive an actuation of an aspect of said SGML-based application interface; and

10   deliver a command to the server corresponding with said actuation; and the server being operative to:

receive the invocation request from the client;

extract client-specific information from the client and the invocation request;

15   prepare an SGML-based application interface based on the client specific information;

transmit the SGML-based application interface to the client; and

receive a command corresponding to an actuation of an aspect of said SGML-based application interface and in response to receiving said command prepare
20   a modified SGML-based application interface and deliver it to the client.
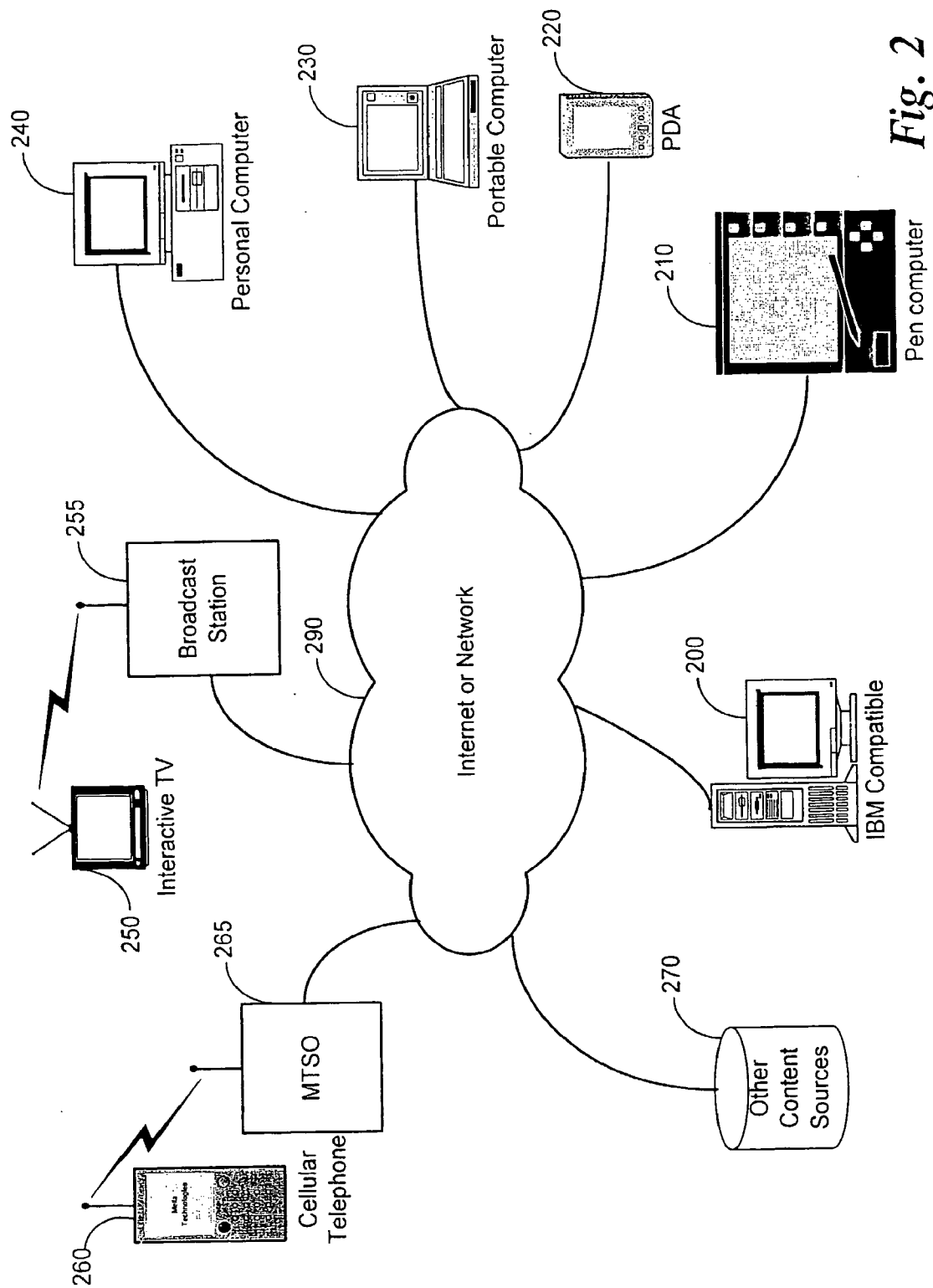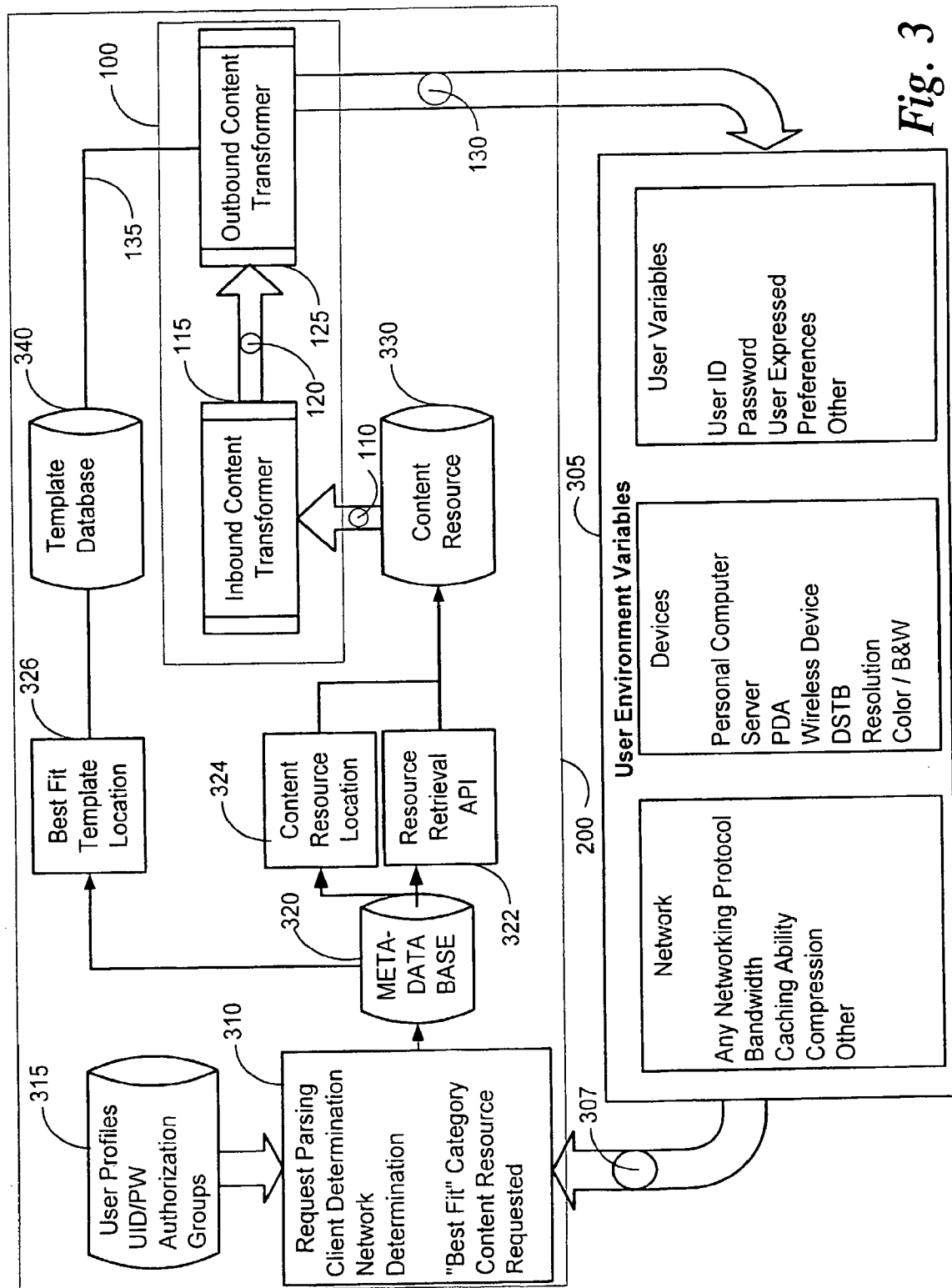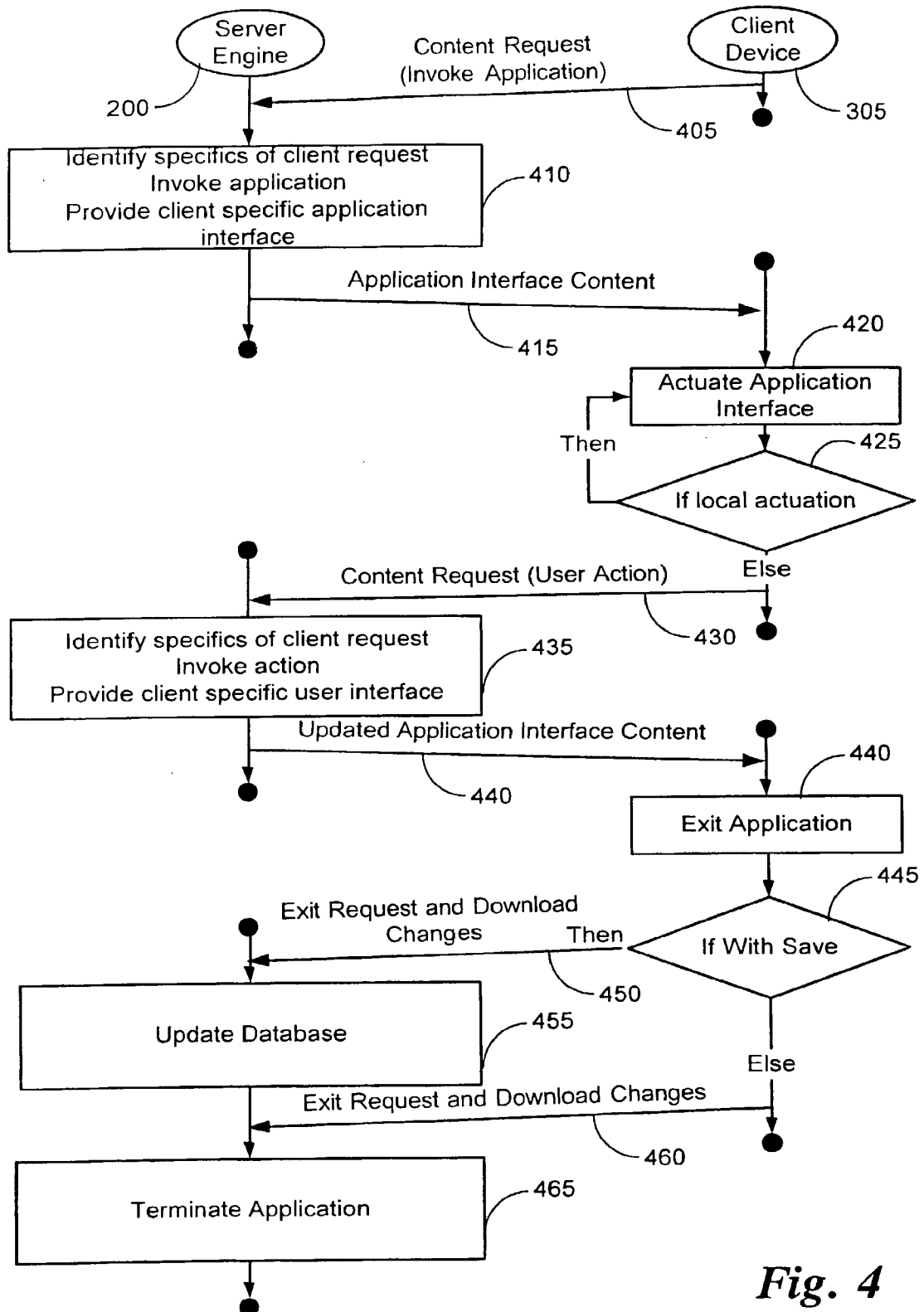
Fig. 1b

Fig. 1a

Personal Computer 240

Portable Computer 230

PDA 220

210 Pen computer

Broadcast Station 255

Interactive TV 250

290 Internet or Network

200 IBM Compatible

265 MTSO

Cellular Telephone

260

270 Other Content Sources

Fig. 2

Fig. 3

Fig. 4